Tampereen yliopisto

Jussi Joenperä

AUTOMATIOSIDUN VARASTON DEMONSTRAATIO LEGO MINDSTORMS EV3 -ROBOTILLA

Tekniikan ja luonnontieteiden tiedekunta
Kandidaatintyö
Joulukuu 2019

TIIVISTELMÄ

JUSSI JOENPERÄ : Automatisoidun varaston demonstraatio Lego Mindstorms EV3-robotilla
Demonstration of an automated warehouse with Lego Mindstorms EV3 robot
Kandidaatintyö
Tampereen yliopisto
Kandidaatintyö, 29 sivua
Automaatiotekniikan koulutusohjelma
Joulukuu 2019
Pääaine: Automaatiotekniikka
Tarkastaja: Mikko Salmenperä

Työssä tutustutaan automatisoidun varaston toteuttamiseen Lego Mindstorms EV-robotin avulla. Tavoitteena on luoda pieni automaatiojärjestelmä, jolla robotti pystyy hoitamaan automatisoidun varaston tehtäviä. Järjestelmän toteutuksessa hyödynnetään automatisoitua varastointi- ja hakujärjestelmää, lyhennettynä ASRS. Ohjelmiston suunnittelussa hyödynnetään UML-malleja. Työn teoriaosuudessa käydään läpi ASRS:n periaatteita, rakennetta ja vaatimuksia. Tämän jälkeen käydään läpi valitun Lego rakennussarjan ohjelmointi- ja rakennusmahdollisuuksia. Näiden tietojen pohjalta toteutetaan automatisoitu varasto. Työn haasteena ovat käytettävien sähkömoottoreiden epätarkkuus ja hitaus sekä automaation vaatimien turvallisuusvaatimusten täyttäminen. Työn lopputuloksena saatiin toimiva varastointijärjestelmä, joka ei kuitenkaan täyttänyt kaikkia sille asetettuja vaatimuksia.

Avainsanat: Automaatio, Python, Lego, Mindstorms, EV3, Automatisoitu varasto, ASRS

SISÄLLYSLUETTELO

LYHENTEET JA MERKINNÄT

ASRS	automatisoitu järjestelmä, jota käytetään tavaran varastointiin ja sen hakemiseen varastosta
EV3	Lego Mindstorms-robottisarjan kolmas versio
ev3dev	Debian Linux-pohjainen käyttöjärjestelmä, jonka avulla Lego Mindstormia voi ohjelmoida halutuilla kielillä
UML	graafinen mallinnuskieli, jossa ohjelmiston rakenne, tilat ja käyttäytyminen kuvataan kaavioiden avulla

1. JOHDANTO

Automatisoidulla varastolla tarkoitetaan itsetoimivaa varastoa, jossa käytetään tietokoneohjattuja laitteita tavaroiden varastointiin ja kuljetukseen. Automatisoitu varastointi- ja hakujärjestelmä ASRS (Automated Storage and Retrieval System) on automatisoitu varastointijärjestelmä. ASRS on levinnyt moniin eri toimialoihin, ja sen käyttö lisääntyy ajan myötä. Levinneisyyden ansiosta siitä löytyy monenlaisia toteutuksia.

Tässä työssä tutkitaan automaatiovaraston periaatteita ja havainnollistetaan valittuja toimintoja Lego Mindstorms EV3-rakennussarjan avulla. Työssä tutustutaan valittuun rakennussarjaan, jonka avulla robotti rakennetaan. Kokeellisen osuuden tavoitteena on toteuttaa hyllykkökäyttöön tarkoitettu varastorobotti ja sille ohjelma. Tässä osuudessa hyödynnetään ASRS:n periaatteita. Työn haasteina ovat rakennussarjan mittausten epätarkkuus ja liikkeiden hitaus sekä automaation vaatima turvallisuustoimenpiteiden saaminen järjestelmään. Työn vaatimuksia ovat robotin rakentaminen Lego-paketin osilla, turvallisuuselementtien lisääminen sekä ohjelman toteuttaminen kolmannen osapuolen tekemän kirjaston avulla varastorobottiin. Sama kolmas osapuoli tarjoaa kirjaston monelle eri ohjelmointikielellä. Tähän työhön on valittu ohjelmointikieleksi tekijälle tuttu Python.

Toisessa luvussa perehdytään automaatiovarastoon. Ensiksi käsitellään automatisoidun varaston perusideaa, jonka jälkeen käsitellään mitä automaation sovellus tässä työssä tulee sisältää. Lopuksi käsitellään automaation turvallisuusvaatimuksia. Kolmannessa luvussa käydään läpi robotin ohjelmistoympäristö. Käydään läpi valittu ympäristö ja miten siihen on päädytty.

Neljännessä luvussa käydään läpi käytännön osuus, joka etenee työn toteutuksen mukaan. Jokaisessa alaluvussa käydään läpi eteen tulleet haasteet ja mahdolliset ratkaisut. Lyhyesti: työn suunnitteluprosessi, toteutus ja haasteet. Samassa luvussa käsitellään työn tuloksia sekä niiden jatkokehitystä. Viides luku kokoaa yhteen tärkeimmät havainnot ja johtopäätökset.

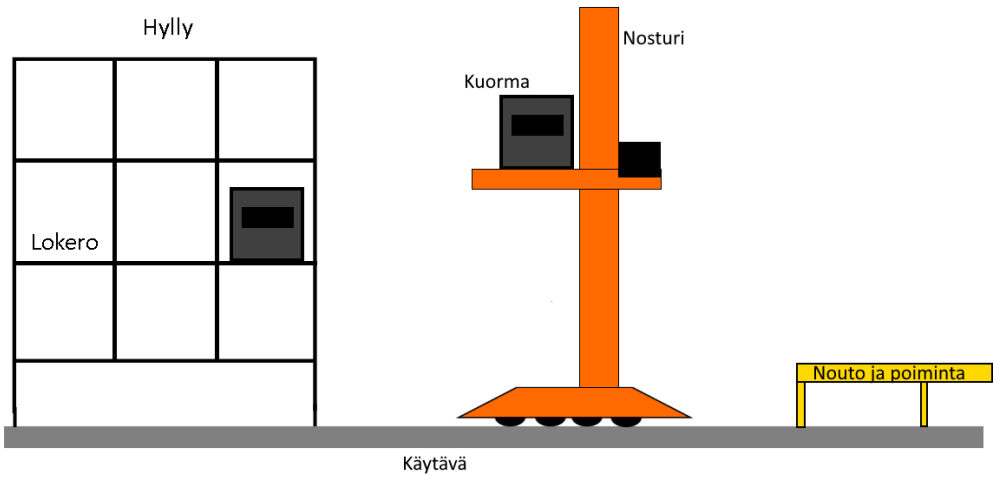
2. AUTOMATISOITU VARASTO

Tässä luvussa perehdytään automatisoituun varastoon (ASRS), sen rakenteeseen ja toimintoihin. Tämän lisäksi tutustutaan tarkemmin varastojen automaatiojärjestelmien periaatteisiin, vaatimuksiin sekä varaston suunnitteluun ja turvallisuusvaatimuksiin. Suunnittelulle ja toteutukselle löytyy erilaisia vaihtoehtoja, joita tässä luvussa käsitellään. Kaikista vaihtoehdoista käydään läpi vain tähän työhön soveltuvia.

2.1 Yleistä ja automaation vaatimukset

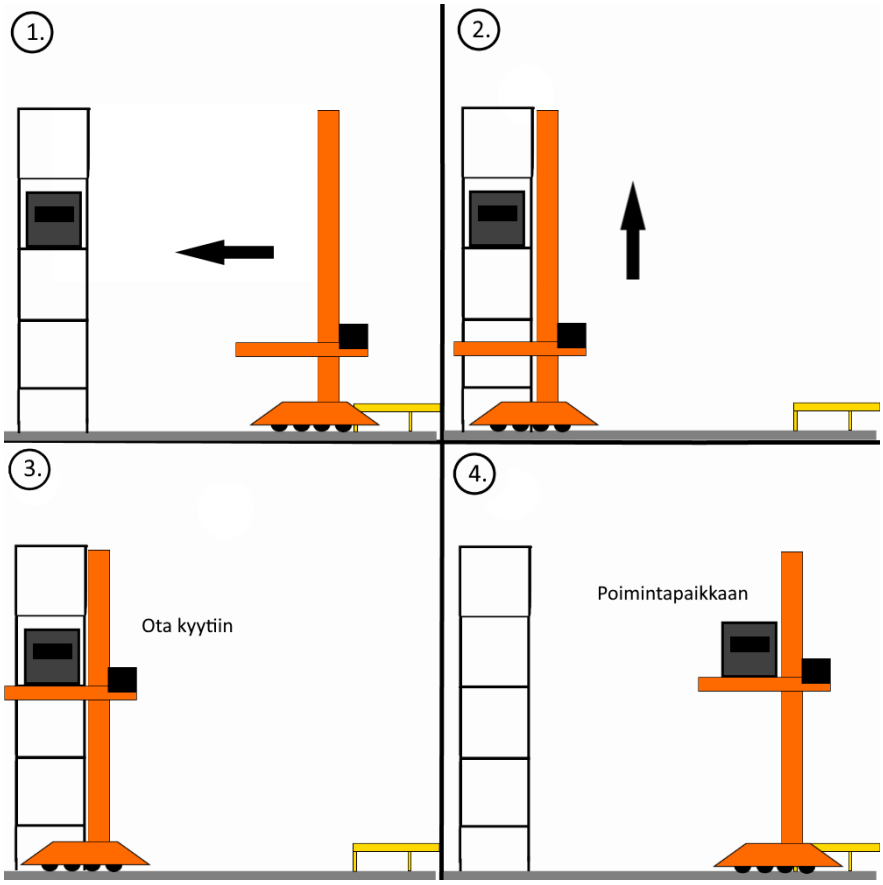
ASRS on varastointijärjestelmä, joka käyttää määrätyn reitin säilytys- ja hakulaitteita. Järjestelmä pystyy käsitellä kuormaa ilman käyttäjän väliintuloa. Tämä tekee järjestelmästä täysin automatisoidun. ASR-järjestelmiä käytetään tavaroiden varastointiin ja niiden hakemiseen varastosta. Järjestelmiä on yleisesti käytetty jakelu- ja tuotantoympäristöissä. Automatisoidulla järjestelmällä saadaan lisättyä luotettavuutta ja vähennettyä virhetasoa. Tämä on tullut ilmi, kun sitä on verrattu ei-automatisoituun järjestelmään. [1]

ASRS:n laitteet toimivat yhdellä tai useammalla kiskolla. Laitteet kulkevat määrätyillä alueilla säilytyshyllyjen välillä. Järjestelmä on yleensä rakennettu telineistä, joita nosturit nostavat hyllyjen välisten käytävien läpi. Järjestelmässä on viisi pääkomponenttia: hyllyt, nosturit, käytävät sekä nouto- ja poimintapaikat. Hylly on paikka, jossa varastoitavaa kuormaa voidaan säilyttää. Nosturi on automatisoitu varastointi- ja hakukone, ja sitä kutsutaan myös varastorobotiksi. Se osaa itsenäisesti noutaa ja laskea kuormaa sekä liikkua. Käytävä on tyhjä tila hyllyjen välillä, missä nosturit voivat liikkua. Noutopaikka on paikka, jonne haettu kuorma viedään ja jonne kuorma tuodaan varastointia varten. Poimintapaikassa työntekijät poistavat yksittäisiä esineitä noudetusta kuormasta ennen kuin kuorma lähetetään takaisin järjestelmään. [1] Alla olevassa kuvassa nähdään esimerkki automatisoidusta varastosta.



Kuva 1. *Esimerkki automatisoidusta varastosta. Kuvassa on yhden yksikön kuormituskäytävällä varustettu ASRS*

ASR-järjestelmät ovat hyvin sopeutuvia kutakin käyttötarkoitusta varten. On olemassa erilaisia vaihtoehtoja nostureiden ja hyllyjen toteutukseen sekä kuorman käsittelyyn. Nosturi voidaan toteuttaa toimimaan joko vain yhden hyllyn äärellä, tai se voi liikkua hyllyjen välillä. Nosturia suunniteltaessa päätetään, kuinka monta kuormaa se pystyy kerralla liikuttamaan. Nosturi voi olla päältä ajettava, jolloin ihminen voi käsitellä kuormaa hyllylokerossa. Poimintapaikkaa tarvitaan vain, jos on tarve ottaa esineitä pois kuormasta. Hylly voidaan toteuttaa niin, että jokaiseen lokeroon mahtuu enintään yksi tai kaksi kuormaa. Hylly voi olla sivuttain liikuteltava, pyörivä tai paikallaan pysyvä. [1] Alla olevassa kuvassa on havainnollistettu tilannetta, jossa nosturi hakee kuorman hyllystä ja vie sen poimintapaikkaan.



Kuva 2. *Nosturi hakee laatikon hyllystä.*

Järjestelmä, jossa jokaisella käytävällä on yksi nosturi, on yleisin toteutus ASR-järjestelmälle. Nosturi ei voi poistua käytävältä, jolle se on määrätty. Tässä tilanteessa ihmisiä ei ole mukana kuorman käsittelyssä. Nosturi voi ottaa kyytiin vain yhden kuorman kerrallaan. Hyllyt pysyvät paikoillaan. Jokaiseen hyllyn lokeroon mahtuu vain yksi kuorma. Jokainen kuorma on suoraan nosturin saatavilla. Tämän tyyppistä järjestelmää kutsutaan nimellä yhden yksikön kuormituskäytävällä varustettu ASRS. [1]

2.2 Suunnittelu ja turvallisuus

Automatisoidun varaston toteutus alkaa sen suunnittelulla. Suunnittelu voidaan jakaa viiteen eri vaiheeseen. Nämä vaiheet ovat varaston yleinen rakenne, osaston asettelu, toimintastrategian valinta, laitteiden valinta ja mitoittaminen. Suunnittelupäätökset vuorovaikuttavat vahvasti toisiinsa, ja niiden välistä rajaa on vaikea määrittää. Näistä syistä mitään suunnittelupäätöstä ei tehdä itsenäisesti, vaan jokainen päätös vaikuttaa kaikkiin muihin. Suunnitteluvaiheessa täytyy ottaa myös huomioon toiminnan suorituskykyyn vaikuttavat tekijät. [2]

Varaston yleisellä rakenteella on vaikutusta toiminnallisiin osastoihin. Yleiseen rakenteeseen kuuluvat muuan muassa varastojen lukumäärä, teknologian taso ja tilausten kokoonpano. Suunnitteluvaiheen tarkoituksena on pyrkiä täyttämään varaston suorituskyvyn vaatimukset. Kuitenkin pyritään minimoimaan kustannukset. [2]

Osaston asettelulla määrätään kuormalavojen pinoamismalli, varasto-osaston asettelu ja ASR-järjestelmän kokoonpano. Tässä työssä kiinnostaa kaksi viimeisintä asiaa. Varasto-osaston suunnittelulla pyritään minimoimaan materiaali- ja rakennuskustannukset. Päätöksiin kuuluvat käytävien lukumäärä sekä leveydet ja korkeudet. ASR-järjestelmän kokoonpanolla on vaikutusta nosturien ja käytävien lukumäärän ja hyllyjen mittoihin. Osaston asetteluvalinnoilla vaikutetaan ylläpitokustannuksiin, varastointikapasiteettiin ja laitteiden käyttöön. [2]

Toimintastrategioilla tarkoitetaan päätöksiä, joilla on suuria vaikutuksia muihin suunnittelupäätöksiin. Kun strategiat on valittu, niitä harvoin jälkeenpäin muutetaan. Varastoon liittyvillä strategioilla voidaan saada pienennettyä kuormien kuljetusaikoja. Valinnoissa käytetään apuna erilaisia simulaatioita varastoista. [2]

Laitteiden valinnoilla määrätään automaatiotaso varastossa. Parhaan automaatiotason saavuttaminen ei ole yksinkertaista useimmissa tapauksissa. Usein suunnittelijoiden ja osastonjohtajien kokemukset päättävät sopivan automaatiotason. Laitteiden valinnassa pitää huomioida varaston tyyppi ja materiaalihallintasysteemien käyttö. Nämä päätökset vaikuttavat suuresti varaston kuluihin ja suorituskykyyn. [2]

Varaston mitoitus jaetaan kahteen osaan. Ensin määritetään varaston koko, joka määrää varastokapasiteetin. Varaston on oltava tarpeeksi tilava, jotta se täyttää tarpeen varastoitavalle tavaralle. Toisena määritetään lattia-ala varastoa varten, jotta pystytään arvioimaan rakennus- ja käyttökustannukset. [2]

Robotti ja ihminen vaarantavat toistensa turvallisuuden työskentelemällä samassa fyysisessä tilassa. Robotteihin on suunniteltu ja toteutettu ratkaisuja turvallisuuden varmistamiseksi. Ratkaisuihin kuuluvat vaaratilanteen havaitseminen ja toimiminen vaaratilanteen estämiseksi. [3]

Turvallisuus teollisuusroboteissa saavutetaan eristämällä robotin työalue ihmisistä. Ihmisen ja robotin työtilan välissä on ovi. Kun ovi avataan, robotti pysäytetään välittömästi. Toisena vaihtoehtona on valoverho, joka havaitsee, kun sen ohi kuljetaan. Heti havaitsemisen jälkeen robotti pysäytetään. Tekstissä mainittu standardi ISO-10218 määrittää enimmäisnopeudet, -tehot ja -voimat roboteille. Näillä rajoituksilla pyritään antaa ihmisille mahdollisuus välttää vaarallisia tilanteita. Robotissa on sensoreita, jotka havaitsevat robottiin aiheutuvan vastuksen. Kun liian suuri vastus huomataan, robotti pysäytetään. [3]

3. LEGO MINDSTORMS EV3 - YMPÄRISTÖ

Tässä luvussa käsitellään Lego Mindstorms-rakenussarjan sisältöä sekä laitteiden teknisiä tietoja, automaation soveltamista Lego Mindstormiin ja valittua ohjelmointiympäristöä. Lego Mindstorms tarjoaa laajat mahdollisuudet robottien rakentamiseen ja ohjelmointiin. Laitteilla on helppo ajaa muokattuja laiteohjelmistoja. Tämän vuoksi Lego Mindstormille voi kirjoittaa omia ohjelmointiympäristöjä.

3.1 Lego Mindstorms

Lego Mindstorms EV3 on Lego Groupin valmistama rakennussarja. Rakennussarja sisältää monenlaisia rakennuspalikoita, keskusyksikön, sähköllä toimivia toimilaitteita, antureita ja johtoja. Rakennuspalikoiden mukana on osia, joita ei yleensä näe muissa Legon rakennussarjoissa. Näillä erikoisilla palikoilla rakennettu robotti voi tehdä erilaisia toimintoja ja liikkeitä. Sarjaan sisältyy kolme moottoria, jotka tuottavat pyörimisliikettä. Yksi näistä moottoreista on muita pienempi. Siitä käytetään nimeä keskisuuri moottori. Keskisuuri moottori on suuria moottoreita tarkempi. Kummankin moottorityypin pyörimissensorit toimivat yhden asteen tarkkuudella. [4]

Sarja sisältää erilaisia antureita. Värianturi tunnistaa seitsemän eri väriä ja mittaa valonvoimakkuutta. Kosketusanturi tunnistaa kolme eri tilaa: kosketus, törmäys ja vapautus. Infrapuna-anturi tunnistaa kohteita lyhyen kantaman päästä. Erikseen myytävällä kaukoohjauksen infrapunavalolla voidaan ohjata robottia. Infrapuna-anturi tunnistaa painetun painikkeen infrapunavalolla. [4]

Rakennussarjan keskusyksikkö EV3 Brick on robotin ohjauskeskus ja tehonlähde. Keskusyksikkö on Linux-pohjainen tietokone, joka sisältää 300 MHz:n prosessorin ja 64 MB RAM:ia. Keskusyksikössä on mustavalkoinen näyttö sekä seitsemän painiketta. Painikkeita ovat nuolinäppäimet, ok- ja takaisinpainikkeet. Keskusyksikössä on neljä input- ja outputporttia. Siinä on myös muita liitäntöjä. Micro USB-portin avulla pystytään muodostamaan yhteys tietokoneen kanssa. USB-isäntäporttiin voidaan kytkeä Wifi-adapteri, tai vaihtoehtoisesti enintään neljä muuta keskusyksikköä. SD-muistikorttipaikka on tarkoitettu lisämuistille. Keskusyksikköön on liitettävissä mikro SD kortti, jonka enimmäismuisti on 32 gigatavua. SD-kortin tuki on hyödyllinen ominaisuus. Tuen avulla keskusyksikön sisäisiä osia ei tarvitse muuttaa, kun ajaa muokattuja laiteohjelmistoja. Keskusyksikköön kuuluu myös sisäänrakennettu kaiutin. [4]

Kuva 3. *Lego Mindstorm EV3 - keskusyksikkö.*

3.2 Python-ohjelmointikieli

Työssä käytetään kolmannen osapuolen tarjoamaa kirjastoa, koska Legon tarjoama graafinen ohjelmointiympäristö on liian alkeellinen tätä robottia varten. Legon ohjelmointiympäristö perustuu lohkoihin, joilla on jokin toiminto. Lohkoja voidaan liittää toisiinsa. Näin saadaan toimiva ohjelma. Legon ohjelmisto tarjoaa eri määrän muuttujia ja tietorakenteita. Ohjelmisto ei tarjoa olio-ohjelmoinnin mahdollisuutta. [5] Työssä tehtävän ohjelman, joka käsittelee varastoa, tulisi käsitellä myös hyllykön tietoja. Legon ohjelmiston tietorakenteet eivät kuitenkaan riitä tähän tarkoitukseen. Tässä työssä on tarkoitus keskittyä korkeamman tason ohjelmointikieliin. Näiden edellä mainittujen asioiden vuoksi Legon ohjelmistoympäristö rajataan pois.

Tutkimuksen yhtenä kiinnostuksen kohteena on toteuttaa ohjelma Pythonilla tai C++:lla. Internetistä löytyi kirjasto nimeltä ev3dev, jonka avulla Lego Mindstormia voi ohjelmoida halutuilla kielillä. Lopulta valittiin Python, koska siitä on tekijällä eniten kokemusta. Valitulle kirjastolle löytyy laaja dokumentaatio juuri tälle ohjelmointikielelle. [6]

Ev3dev-kirjasto on Debian Linux-pohjainen käyttöjärjestelmään. Kirjasto toimii monilla alustoilla, jotka ovat yhteensopivia Lego Mindstormin kanssa. Näihin alustoihin kuuluu työssä käytettävä Lego Minstorms EV3. Kirjasto tarjoaa matalan tason ohjaimet Lego Mindstorm-laitteiden ohjaamiseen. [7]

Kirjasto tarjoaa kattavat rajapinnat antureille, moottoreille ja keskusyksikölle. Keskusyksikön painikkeiden käsittely voidaan ohjelmoida kolmella eri tavalla. Ensimmäisellä tavalla jokaiseen painikkeeseen liittyy tapahtumakäsittelijä. Käsittelijää kutsutaan, kun sitä vastaavaan painikkeen tila muuttuu. Painikkeen tiloja on kaksi: painettu ja vapautettu. Toisella tavalla keskusyksikön funktio palauttaa listan painikkeista, joita painetaan funktion kutsuhetkellä. Kolmannella ohjelmointitavalla ohjelmaa käsketään odottamaan, kunnes haluttua painiketta tai painikkeiden yhdistelmää painetaan. Tähän toimintoon voidaan asettaa aikakatkaisu. Jos määrättyjä painikkeita ei paineta tähän aikaan mennessä, ohjelma jatkaa etenemistään, eikä enää odota painikesyötettä.

Moottoreiden asentoja, pyörimisnopeuksia ja pysäytystoimintoja voidaan muuttaa. Rajapinta tarjoaa viisi funktiota, joilla voidaan pyörittää moottoreita. Ensimmäisenä on funktio, joka käskee moottoria pyörimään, kunnes ohjelma komentaa muuta. Toisin sanoen moottori voidaan asettaa pyörimään ikuisesti. Muut funktiot voidaan selittää lyhyesti. Moottori pyörii määrätyn kulman verran, määrättyyn kulmaan, määrätyn ajan tai määrätyn kierroslukumäärän verran.

Pysäytystoiminnat määräävät, mitä moottori tekee sen pysähtyessä. Pysäytystoimintoja ovat vapaa, jarru ja pito. Vapaa-toiminnalla moottorista poistetaan virta ja se pysähtyy vapaasti. Jarru-toiminnalla moottorista poistetaan virta ja sille asetetaan passiivinen sähkökuorma. Tällä moottori pysähtyy muita toimintoja nopeammin. Pito-toiminta ei poista virtaa moottorista. Sen sijaan toiminta pyrkii pitämään moottorin asennossa, johon se käskettiin pysähtyä. Moottori liikkuu takaisin säilyttääkseen asemansa, jos ulkoinen voima yrittää kääntää sitä. [8]

Kirjasto voidaan ladata verkkosivulta. Samalta sivustolta löytyy ohjeet ohjelmiston asentamiseen sekä miten muodostetaan yhteys keskusyksikön ja tietokoneen välillä. [7] Ladattu ohjelmisto kopioidaan SD-kortille, jotta ohjelmisto saadaan siirrettyä keskusyksikölle. Kortti pitää ensin alustaa oikeaan muotoon. Alustus tapahtuu ohjelmalla, joka löytyy aiemmin mainituista ohjeista. Kortin on oltava keskusyksikön SD-muistikorttipaikassa, kun keskusyksikkö käynnistetään. Keskusyksikkö käynnistää Legon alkuperäisen ohjelmointiympäristön, kun muistikortti poistetaan. Ohjeisiin sisältyy SSH- ja Bluetooth-yhteyksien muodostamiset. Ohjeissa ei selitetty tarkkaan yhteyden muodostumista tai niiden käytöstä. Internetistä etsittiin tarkempia ohjeita tähän ongelmaan. Löytyi opetusvideoita, jotka ohjeistivat askel kerrallaan yhteyden muodostamisen. [9] Samoissa opetusvideoissa neuvottiin, miten pystyy muodostamaan Bluetooth-yhteyden keskusyksikön ja tietokoneen välillä. Vaadittiin Microsoftin Visual Studio Code. Tämä on Pythonille sopiva ohjelmointiympäristö. Visual Studio Codessa pitää asentaa laajennus, jonka avulla tietokone muodostaa keskusyksikköön yhteyden Bluetoothin välityksellä.

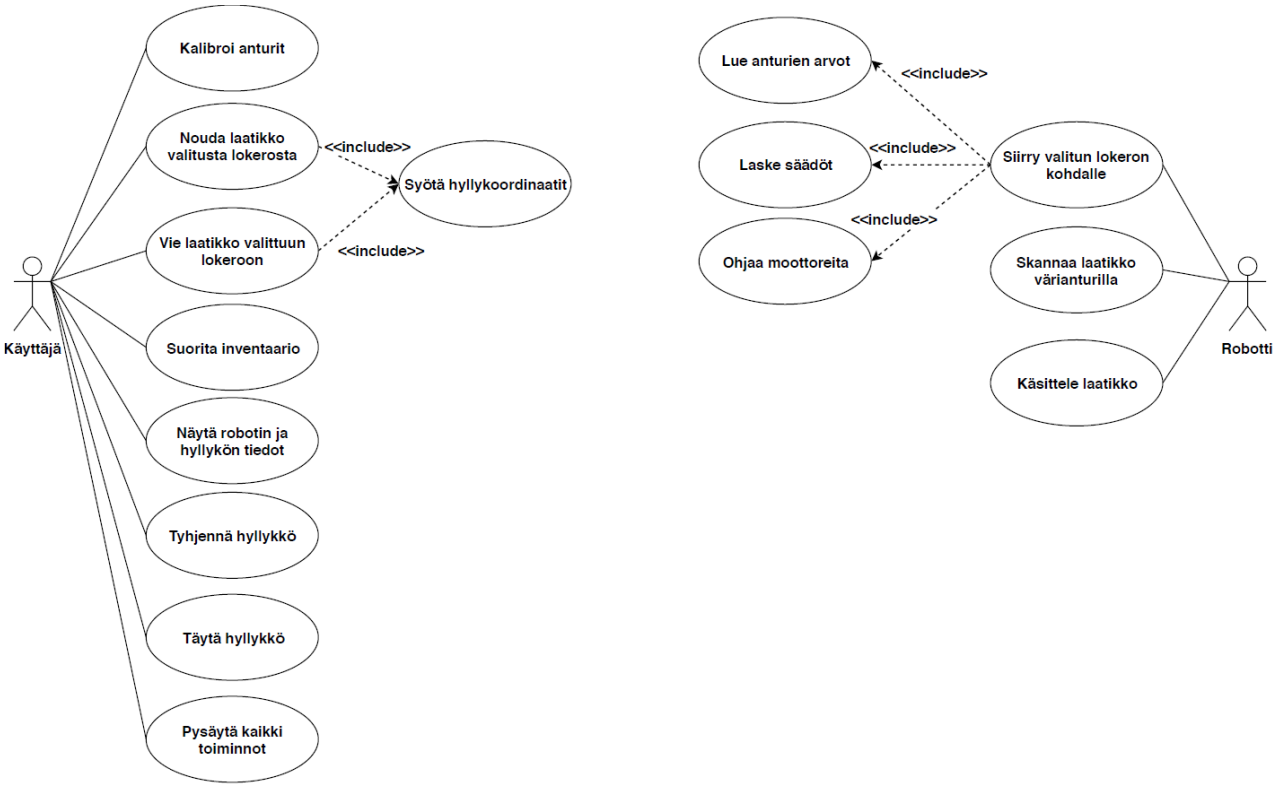
4. AUTOMATISOITU VARASTO LEGO MINDSTORMSILLA

Työssä tarkastellaan, miten Lego Mindstorms-robotilla voidaan suorittaa ASR-järjestelmän toimintoja. Aluksi määritetään vaatimukset varastointijärjestelmälle. Toteutettavalle järjestelmälle tulee olemaan samoja vaatimuksia kuin ASR-järjestelmällä. Vaatimuksissa otetaan huomioon Lego Mindstormsin ja ev3dev:in rajoitteet. Suunnittelu ja toteutus jaetaan kahtia fyysiseen ja ohjelmistoon. Fyysiseen osaan kuuluu hyllykön ja robotin rakentaminen. Työn toteutuksen jälkeen käydään läpi työn aikana ilmenneitä haasteita ja miten niitä saatiin ratkaistua. Lopuksi pohditaan sovelluksen onnistumista, tavoitteiden täyttymistä ja jatkokehitysideoita.

4.1 Järjestelmän vaatimukset

Robotti eli nosturi sisältää samoja toimintoja kuin ASRS. Robotti voi sijoittaa laatikon käyttäjän määräämään lokeroon, kunhan kyseinen lokero on vapaa. Laatikon voi poistaa hyllyköstä, kun käyttäjä syöttää laatikon hyllykoordinaatit. Hyllykön voi täyttää laatikoilla sekä hyllykön voi tyhjentää kokonaan yhdellä komennolla. Robotti voi suorittaa hyllyköstä inventaarion, jolloin robotti käy itsestään koko hyllykön läpi. Robotti tunnistaa laatikon tämän värin perusteella. Väritunnistusta käytetään aina, kun käsitellään laatikoita. Ohjelma näyttää käyttäjälle robotin ja hyllykön tiedot. Robotti voi kalibroida itsensä. Viimeinen komento lisättiin, koska Legon antamien ohjeiden mukaan moottorit unohtavat asemansa ohjelman sulkeutuessa. Robotti suorittaa kaikki nämä toiminnot käyttäjän antamasta käskystä.

Kuvassa 4 näkyy käyttäjän ja robotin käyttötapaukset.



Kuva 4. *Käyttötapauskaavio*

Ohjelmointikielenä käytetään Pythonia ja suunnittelun määrittelykielenä UML:ää. Järjestelmää pystyy käyttämään vain yksi käyttäjä kerrallaan. Robotin voi siirtää varastosta toiseen. Tämän vuoksi robotti tulee suunnitella siten, että se on helppo purkaa osakokonaisuuksiin ja helppo koota takaisin. Myös tietojen käsittely toiminnan onnistumisesta, robotin tilasta ja hyllykössä olevien laatikoiden tiedoista tapahtuvat automaattisesti. Kuljetettavien laatikoiden on oltava tarpeeksi kevyitä, jotta moottorit eivät luule niitä ulkoiseksi vastukseksi. Käyttöliittymä on selkeä, eikä sen käyttöön tarvitse erillisiä ohjeita. Ohjelmaan on helppo lisätä uusia ominaisuuksia.

Automaatio asettaa järjestelmälle turvallisuusvaatimuksia. Näitä vaatimuksia ovat hätäpysäytystoiminta ja hälytykset. Automaation vaatimukset vaativat toimiakseen eri määrän antureita ja toimilaitteita. Hälytykset ilmoitetaan käyttöliittymässä ja keskusyksikön vilkkuvilla valoilla. Törmäyksenestoa varten tarvittaisiin lisää antureita. Rakennussarjassa ei ole tarpeeksi antureita törmäyksenestoa varten. Tässä työssä ainoastaan moottorit havaitsevat robottiin kohdistuvan vastuksen.

Hätäpysäytyspainike valittiin tärkeimmäksi turvallisuusvaatimukseksi. Robotti lopettaa toimintansa heti, kun hätäpysäytyspainiketta painetaan. Painike saadaan toteutettua kosketusanturilla. Robotti kertoo käyttäjälle, onko sen hetkinen toiminto jouduttu keskeyttämään. Robotti pystyy jatkamaan toimintaansa pysäytyksen jälkeen. Varasto ja käyttäjä tulee erottaa toisistaan turvallisuuden takaamiseksi.

4.2 Suunnitteluprosessi

Suunnitteluprosessi alkoi siitä, kun Lego Mindstorm rakennussarja saatiin yliopistolta. Rakennussarjaan ja kolmannen osapuolen kirjastoon tutustuttiin rakentamalla ja ohjelmoimalla pieni koerobotti. Alkuperäisenä ajatuksena oli tehdä robotti, joka liikkuu rajatulla alueella ja siirtää laatikoita paikasta toiseen. Päädyttiin kuitenkin tekemään robotti, joka hallinnoi yhtä määrättyä hyllyä. Tähän tarkoitukseen sopii teoriaosuudessa mainittu yhden yksikön kuormituskäytävällä varustettu ASR-järjestelmä. Suunnitteluun käytettiin löyhästi teoriaosuudessa mainittuja suunnitteluvaiheita, koska automatisoidusta varastosta tehdään melko yksinkertainen. Turvallisuusvaatimukset otetaan huomioon.

Internetistä etsittiin mahdollisia malleja robottia varten. Internetistä löytyy paljon harrastelijoiden tekemiä valmiita malleja. Löytyi Lego Mindstormia käyttävä varastorobotti. Samalta sivulta löytyi videolinkki, jossa näkyi kyseisen robotin toimintoja. Näistä tiedoista otettiin mallia tähän työhön. Sivulta löytyi myös hyllykön malli, joka on sopiva tälle robotille. [10]

Suunnittelu ja toteutus eivät olleet tarkalleen erillisiä prosesseja. Kumpaakin tehtiin limittäin. Iso osa työstä eteni samaa rataa toteutuksen kanssa. Robotin suunnittelussa ei käytetty minkäänlaista työkalua, vaan lähdettiin kokoamaan robottia yksi osa kerrallaan. Käytetään ketterää kehitystä. Robotin suunnittelussa piti ottaa huomioon rakennussarjan ja moottoreiden rajoitukset. Hyllykön suunnittelussa rajoittavin tekijä oli sen korkeus. Tätä rajoitti robotin pystyliikkeen enimmäiskorkeus. Ohjelmiston suunnittelu lähti liikkeelle komennoista, joita käyttäjä voi antaa ohjelmalle. Näistä saatiin suunniteltua ohjelman käyttöliittymä. Käyttöliittymässä piti ottaa huomioon käyttäjän antamat komennot keskusyksikön painikkeiden avulla. Aikaisemmin mainitut vaatimukset määrittivät robotin tärkeimmät toiminnat.

4.2.1 Robotin ja hyllykön suunnittelu

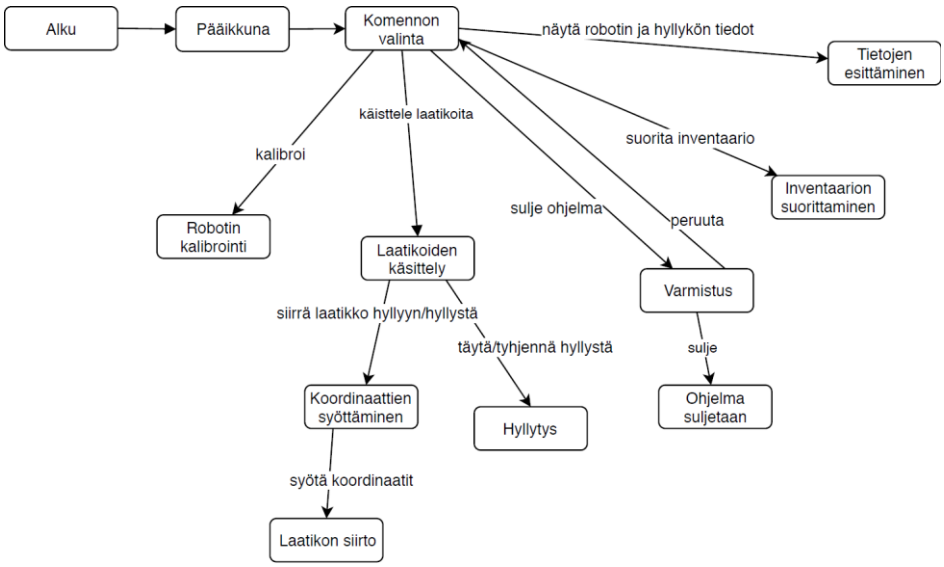
Rakennussarjan moottoreita testattiin pienen koerobotin avulla. Koerobotti ohjelmointiin liikkumaan siten, että ensin se pyrähti ympäri, liikkui eteenpäin, pyörähti 180 astetta ja lopuksi puhui pari sanaa suomea. Lego Mindstormin ohjeissa kerrottiin, että moottorit eivät ole kovinkaan tehokkaita. Tämä ilmeni myös testatessa. Myös moottorien anturit ovat epätarkkoja. Nämä piti ottaa huomioon varsinaista robottia rakentaessa. Lego Mindstorm EV3 pakkauksen mukana yli 500 rakennuspalaa sekä erilaisia antureita ja sähkömoottoreita. Näitä käytetään robotin rakentamisessa. Liukuhihnan avulla robotti pystyy ottamaan yhden laatikon kerrallaan kyytiin ja pois. Iso moottori, joka hoitaa robotin pystyliikkeen, ei pysty siirtämään suuria massoja. Sama moottori liikuttaa robotin liukuhihnaosuutta, joten sen päälle ei voi asettaa suurta taakkaa.

Kaksi isoa moottoria hoitaa sivuttais- ja pystyliikkeet. Keskisuuri moottori liikuttaa liukuhihnaa. Värisensori lukee laatikon tiedot. Kosketusanturi toimii hätäpysäytyspainikkeena. Robotin pystyliikkeen kanssa oli ongelmia, sillä legopakkauksen mukana ei tullut osia, joilla tämä olisi pystynyt saavuttaa. Piti erikseen hankkia suoria hammasrataspalkkeja ja reikäpalikoita. Näillä palikoilla ja pyöreillä hammasrattailla saadaan robotin tarvitsema pystyliike.

Hyllykkö suunniteltiin niin, että siinä tulee olemaan neljä tasoa. Jokaisessa tasossa on kolme lokeroa. Hyllykön rakentamisessa käytetään Lego Duplo-palikoita. Hyllykön vasempaan reunaan, aivan alimman rivin vasemmalla puolella, tulee paikka, jossa liukuhihnan moottori voidaan kalibroida. Saman rivin oikeaan reunaan tulee ylimääräinen lokero vienti- ja tuontipaikka laatikoille. Duplo-paloja käytetään myös laatikkoina, joita varastoidaan hyllykköön. Rautalangasta tehdään kahvat laatikoihin. Näiden avulla robotti pystyy liikuttaman laatikoita. Hyllykön lokeroista tulee tarpeeksi syviä. Näin niihin mahtuu koko laatikko. Lisäksi lokeroihin tehdään niiden perälle este, jotta laatikon kahva jää lokeron ulkopuolelle.

4.2.2 Ohjelmiston suunnittelu

Navigointikaaviosta nähdään mihin toimintoihin käyttäjä päätyy tehdessään valintoja. Kun päästään toiminnan loppuun, ohjelma palaa takaisin pääikkunaan.



Kuva 5. *Navigointikaavio*

Ohjelmiston suunnittelun apuna käytettiin internetistä löytyviä ohjeita [11]. Sivustolla kerrottiin Lego Mindstroms-laitteiden ohjelmointimahdollisuuksista pienten esimerkkien avulla. Näillä esimerkeillä otettiin huomioon ohjelmoinnin mahdollisuudet ja rajoitukset.

Ohjelma on jaettu viiteen luokkaan: Varasto_main, Hyllykkö, Painikkeet, Robotti ja Hätästop.

Varasto_main on luokka, joka hoitaa ohjelman komennot ja tietojen esittämisen käyttäjälle. Luokka tarjoaa eri valikoita, joista käyttäjä voi valita haluamansa komennon. Luokka tulostaa tietokoneen näytölle mitä valintoja käyttäjällä on käytettävissä ja mitä painiketta painamalla valintoja pystyy valitsemaan. Luokka myös tulostaa näytölle tietoa robotin sen hetkisestä tilanteesta ja hyllykön muutoksista. Päävalikosta pystyy sulkemaan ohjelman, suorittaa hyllykön inventaario, suorittaa robotin kalibrointi, tulostamaan robotin ja hyllykön tiedot tai menemään laatikoita käsittelevään valikkoon. Valikosta, jossa käsitellään laatikoita, voi valita laatikon lisäämisen hyllyköön ja poistamisen hyllyköstä sekä hyllykön täydentämisen ja tyhjentämisen.

Hyllykkö-luokka pitää sisällään tiedot hyllykköön säilötyistä laatikoista. Laatikoista tiedetään niiden väri ja minne ne on asetettu hyllykössä. Hyllypaikkaa merkataan rs-koordinaatistolla, jossa r on hyllyrivi alhaalta lähtien ja s on hyllysarake vasemmalta lähtien. Luokassa käytetään listoja listan sisällä, muodostaen taulukon. Lokero saadaan indeksoimalla tätä tietorakennetta käyttäen rs-koordinaatteja. Luokkaa päivitetään Varasto_mainin avulla. Luokan tietoja voidaan muuttaa. Luokka voi myös palauttaa valittuja tietoja, kuten laatikoiden paikat hyllykössä. Värisensori palauttaa mitatun värin englanninkielisenä, ja laatikon väri tallennetaan merkkijonona oikean lokeron kohdalle.

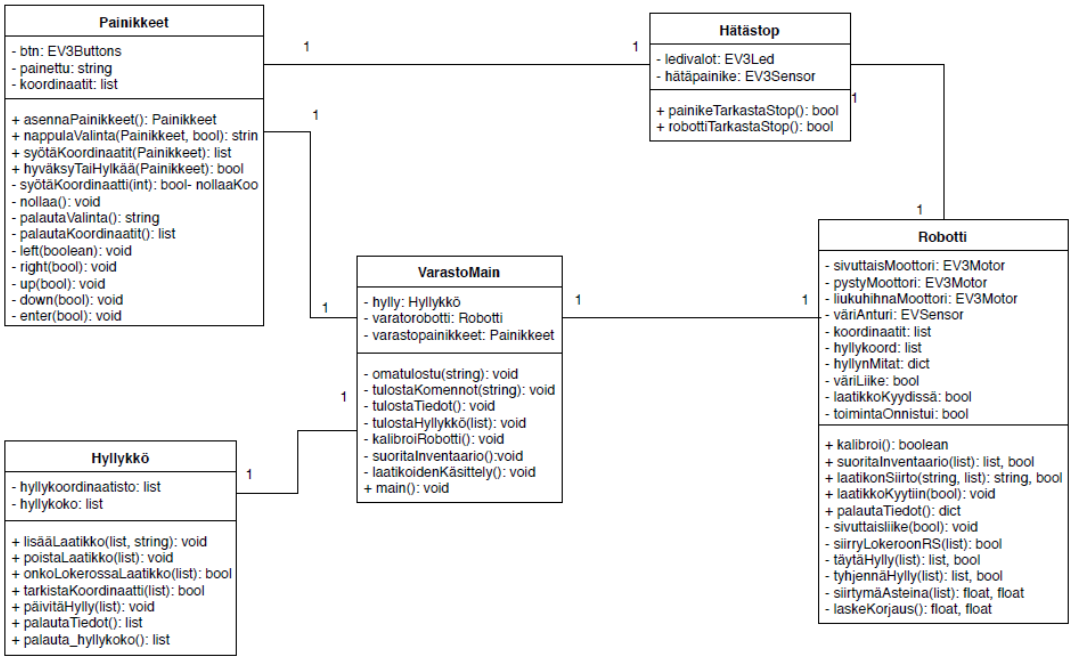
Painikkeet-luokka huolehtii mitä keskusyksikön painikkeita painetaan. Näin käyttäjän antamat tiedot välittyvät Varasto_mainille. Robotti-luokka liikuttaa robottia ja mittaa laatikoiden värit Varasto_mainin käskystä. Luokkaan on ohjelmoitu funktioita, joidenka avulla robotti toimii automaattisesti, kun sille on annettu komento. Painikkeilla annetaan ohjelmalle käskyjä tämän ollessa käynnissä. Ohjelma ottaa käskyjä vastaan, kun robotti ei ole toiminnassa. Hätäpysäytyspainiketta pystyy käyttämään jokaisessa tilanteessa. Ohjelman voi sulkea joko keskusyksikön takaisinpainiketta painamalla tai ohjelman oman komennon avulla. Päävalikossa pystyy valitsemaan toiminnan ’sulje ohjelma’. Ohjelma varmistaa käyttäjältä tämän valinnan. Koska painikkeita on rajallinen määrä, komennot on jaettava eri valikkoihin.

Testatessa moottorien komentoja huomattiin, että moottoreita käskevät pyörimiskomennot voidaan jakaa kahteen osaan. Ensimmäisenä ovat komennot, jotka pysäyttävät ohjelman kulun moottorin liikkeen ajaksi. Näillä komennoilla voidaan liikuttaa vain yhtä moottoria kerrallaan. Toisena ovat komennot, jotka eivät pysäytä ohjelman kulkua robotin liikkuessa. Ohjelmisto tarjoaa moottoreille erillisiä komentoja, jotka pysäyttävät ohjelman kulun robotin liikkumisen ajaksi. Tämän avulla voidaan käskeä kahta tai useampaa moottoria liikkumaan samaan aikaan sekä pysäyttää ohjelman kulku, kunnes moottorit pysähtyvät. Moottorien liikkuessa voidaan tarkistaa jatkuvalla silmukalla hätäpysäy-

tyspainikkeen toimintaa. Testatessa huomattiin myös, että moottorit eivät aina pyöri käsketyn määrän mukaan. Tämän takia ohjelma laskee korjausasteet moottoreille, kun niiden kääntymäasteita määritetään.

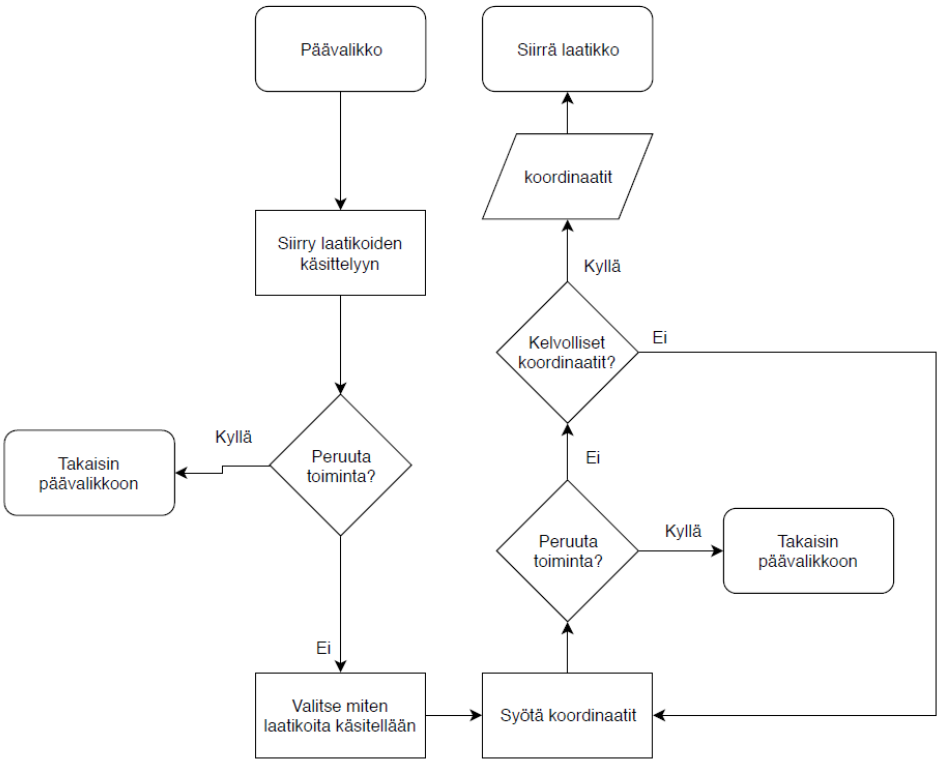
Hätästop-luokka huolehtii hätäpysäytyksestä. Hätäpysäytyspainiketta painamalla robotin toiminta pysähtyy. Hätäpysäytyksen jälkeen robotin voi käynnistää uudelleen, kun keskusyksikön keskimmäistä painiketta painetaan kerran. Ohjelma palaa päävalikkoon ja hyllykön tiedot päivitetään jo kerätyillä tiedoilla.

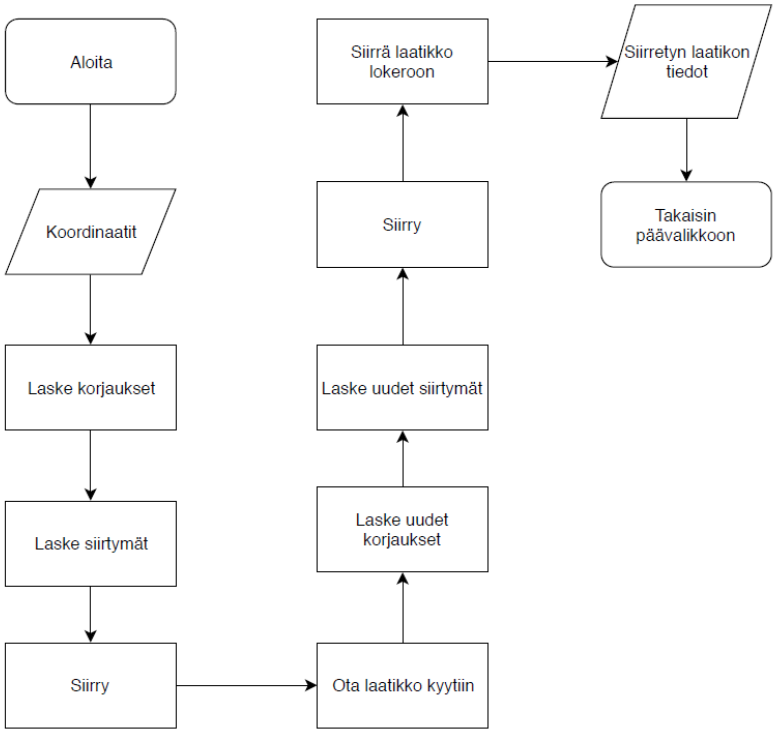
Luokkakaaviosta näkyy ohjelman rakenne sekä ohjelman koostumus eri osista. Luokista näkyy niiden attribuutit ja funktiot.



Kuva 6. *Luokkakaavio*

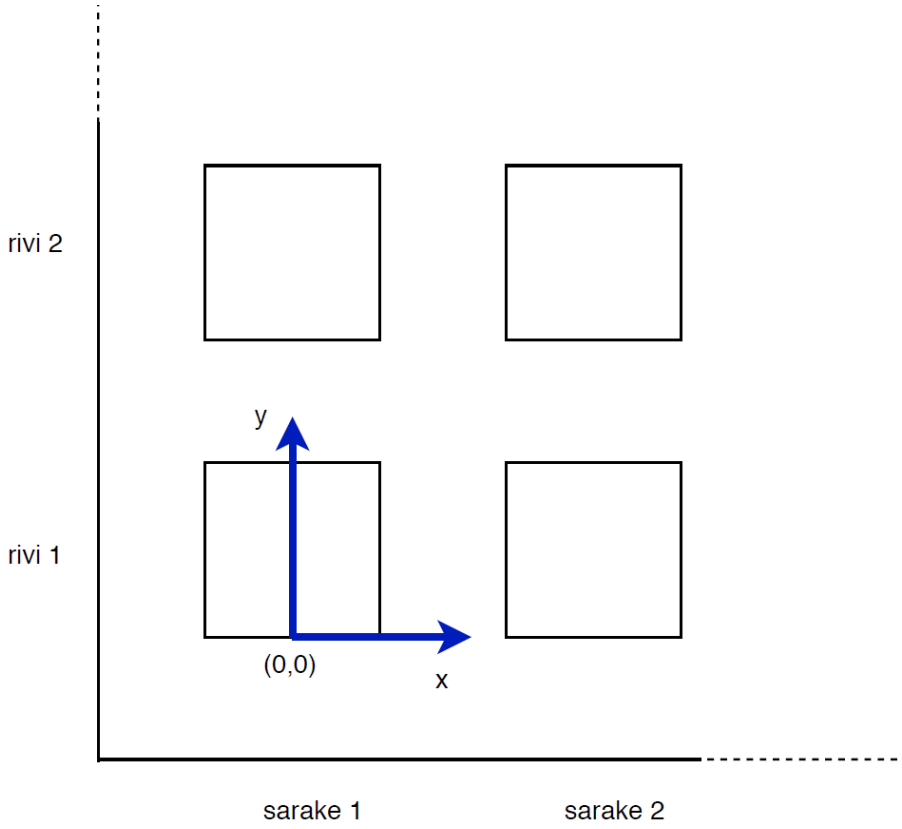
Vuokaavioista nähdään laatikon siirron eri vaiheet. Laatikon valitseminen ja sen siirto on piirretty eri kaavioihin. Ylemmässä kaaviossa näkyy käyttäjän tekemät valinnat. Alemmassa näkyy robotin vaiheet laatikon siirtämiseksi.





Kuva 7. *Vuokaaviot laatikon siirtämisestä*

Kirjaston tutustumisessa huomattiin, että ohjelman ollessa käynnissä tietokoneelta ei pysty antamaa komentoja keskusyksikölle. Päädyttiin, että komennot annetaan keskusyksikön painikkeiden avulla. Myöskään graafista käyttöliittymää ei ole mahdollista käyttää, joten päädyttiin tilanteeseen, jossa ohjelma antaa käyttäjälle tietoa tekstin muodossa tietokoneen näytölle. Keskusyksikön näytölle pystyy tulostamaan tekstiä, mutta päädyttiin tietokoneen näyttöön. Oli helpompaa toteuttaa tulostukset tietokoneen näytöllä kuin keskusyksikön.



Kuva 8. *Suunnittelukuva rs- ja xy-koordinaatioista. Siniset nuolet kuvaavat xy-koordinaatistoa*

Robotin liikkeissä käytetään xy-koordinaatistoa, joka on leveys-korkeus-koordinaatisto. Tässä koordinaatistossa käytetään mittayksikköinä senttimetrejä. Koordinaatiston origo sijaitsee tietyn lokeron alareunan keskellä. Tämä lokero sijaitsee alimmalla rivillä, hyllykön vasemmassa päässä. Hyllykön lokeroiden mitat ja etäisyydet toisistaan pitää mitata. Pitää myös laskea kuinka monta astetta moottorien tulee liikkua jokaista liikuteltavaa senttimetriä kohden. Näillä tiedoilla pystytään muuttamaan rs-koordinaatiston pisteet xy-koordinaatiston pisteiksi. Eli kun halutaan robotin menevän tietyn lokeron kohdalle, robotti osaa laskea tarvittavat liikkeet moottoreille. Robotti käyttää moottorien liikuttamiseen kahta erilaista komentoa. Sivuttaissuunnan kalibrointiin käytetään ajastettua pyörimistä. Kaikkiin muihin liikkeisiin käytetään komentoa, jonka parametrina on moottorin asennon muutos.

Robotin kalibroinnissa robotti ensiksi liikkuu vasemmalle noin kymmenen sekunnin ajan. Robotti kohtaa esteen, joka estää robotin etenemisen. Este sijoitettu niin, että robotin liukuhihna päätyy kalibrointipaikan kohdalle. Seuraavaksi pystyliikkeen moottori liikkuu ylöspäin ja sitten hiukan alaspäin. Moottori poistaa lukituksen ja moottori liikkuu alaspäin oman painonsa takia tornin pohjalle. Moottori liikkuu hiukan ylöspäin hyllykön alimman rivin kohdalle. Moottori nollataan. Liukuhihna liikkuu eteenpäin, kunnes se kohtaa vastuksen. Seuraavaksi liukuhihnaa liikkuu vastakkaiseen suuntaan, kunnes se kohtaa vastuksen. Liukuhihna liikkuu takaisin määrätylle paikalleen. Keskisuuri moottori nollataan. Lopuksi robotti liikkuu hiukan oikealle ensimmäisen hyllysarakkeen kohdalle. Sivusuunnan moottori nollataan.

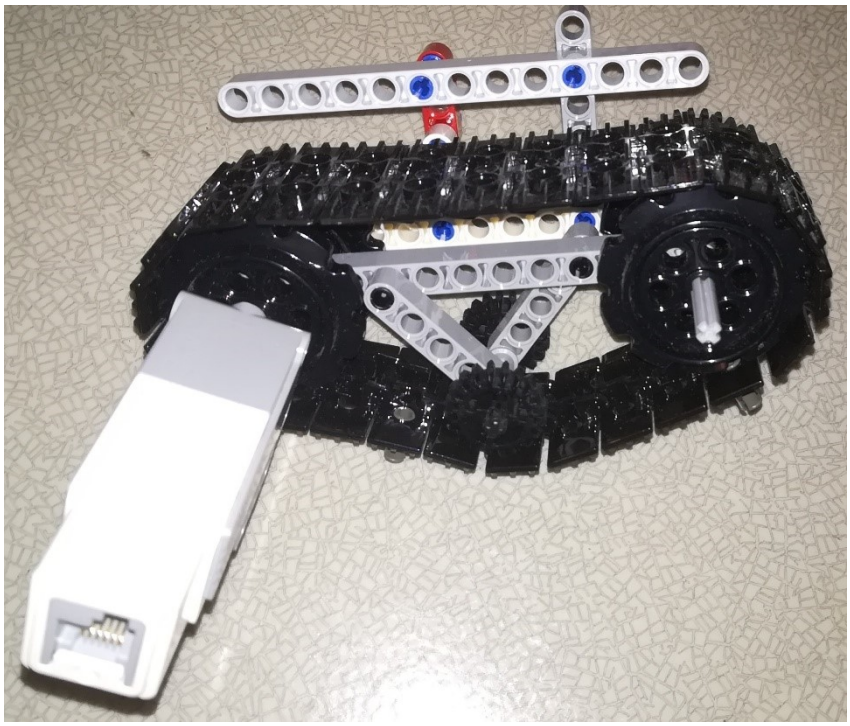
4.3 Työn toteutus

Työn suunnittelu on osittain päällekkäin toteutuksen kanssa. Toteutuksessa käytettiin ketterää menetelmää, scrum. Priorisoitiin toteutuksen kehitysjärjestys, jotta saadaan tehtyä kriittisimmät ominaisuudet. Tärkeimpiä olivat hyllykkö ja robotti. Seuraavaksi tuli ohjelman eri osat. Ensiksi käyttöliittymä ja toisena komentojen syöttö. Näihin kuuluvat luokat Varasto_main ja Painikkeet. Kolmantena oli Hyllykkö-olio. Neljäntenä oli Robotti-olio. Robotti-olion toiminto voidaan priorisoida, koska se on muihin luokkiin verrattuna varsin suuri ja aikaa vievä. Viimeisenä oli Hätästop-luokka. Jokaisen vaiheen edetessä huomattiin parannuksien ja korjausten tarve muissa luokissa. Tästä syystä laadittu kehitysjärjestys ei pysynyt suunnitellussa muodossa.

Ohjelmaa, robottia ja hyllykköä tehtiin rinnan. Tämä tehtiin siitä syystä, koska ohjelma tarvitsi robotin ja hyllykön tietoja pystyäkseen käskemään robotin toimivaan halutulla tavalla. Myös testauksessa tarvittiin kaikkia fyysisiä osia, jotta saatiin tietoon ohjelman tarvittavat korjaukset. Toteutuksen aikana huomattiin monia puutteita ja ongelmia robotin ja ohjelman kanssa. Hyvä osa saatiin ratkaistua. Rakennussarjan palikkoja, kiinnitysosia ym. käytettiin robotin rakentamisessa jokaisen osion kohdalla. Hyllykön ja laatikoiden rakentamisessa käytettiin Duplo-palikoita, teippiä, rautalankaa, pahvia ja paperia.

4.3.1 Robotin ja hyllykön toteutus

Keskusyksikkö toimii akun varassa, mutta rakennussarjan mukana ei tullut laturia. Piti hankkia keskusyksikköön sopivat paristot. Aloitettiin liukuhihnaosuuden rakentamisella. Sen pyörimistä testattiin ilman moottoria. Ensiksi piti arvioida laatikon koko, jotta osattaisiin rakentaa tarpeeksi pitkä liukuhihna. Pyrittiin tehdä liukuhihnaosuudesta mahdollisimman kevyen, jotta nostoliikkeen moottori jaksaa nostaa sitä. Rakennuspaketissa oli liukuhihnan osia. Aloitettiin testaamalla liukuhihnaa. Pienillä palikoilla kokeiltiin liukuhihnan tiukkuus. Liukuhihnaosuuden tukirakenteiden muokkauksella saatiin se sopivan kireäksi. Keskisuuri moottori liikuttaa liukuhihnaa.



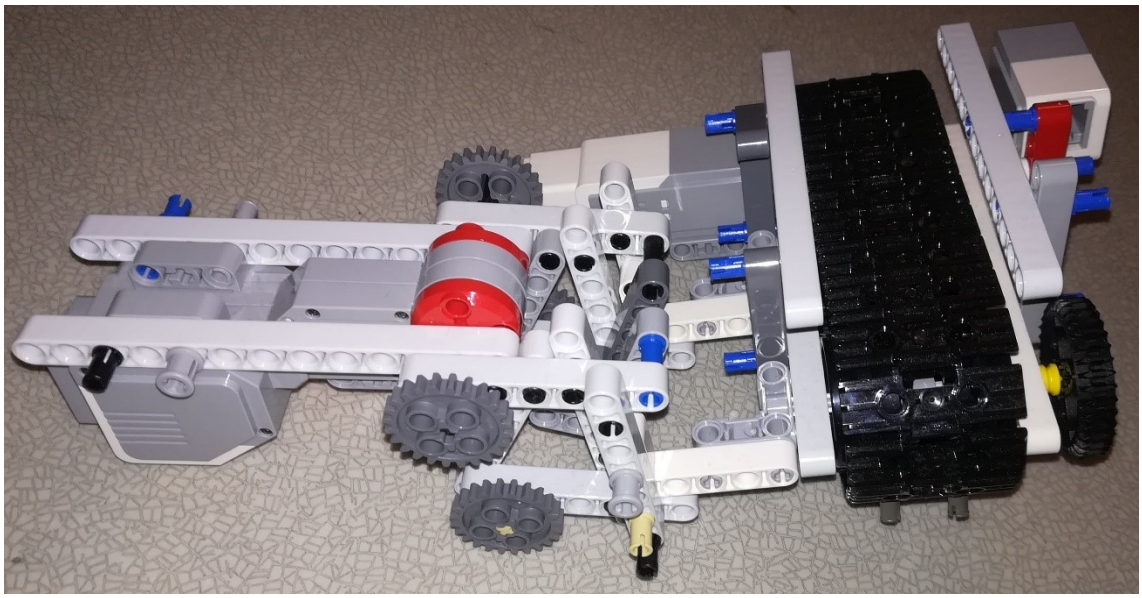
Kuva 9. *Robotin liukuhihnaosuus.*

Liukuhihnan testauksen jälkeen alettiin suunnittelemaan ja rakentamaan robotin torniosuutta. Torniosuus muodostuu neljästä pylväästä, jotka liitettiin yhteen. Aluksi rakennettiin kaksi pylvästä, joissa oli kiinnitettynä hammasratas palkkeja ja niiden kiinnityspalkki. Testattiin, miten saadaan moottori kiinnitettyä jo rakennettuihin pylväisiin. Kaksi hammasratasta kiinnitettiin moottoriin. Hammasrattaat pyörivät moottorin mukana ja niiden avulla saadaan liike. Jotta hammasrattaat pysyvät kiinni hammasratas palkeissa, oli moottoriin kiinnitettävä vastakappale, joka pitää moottorin kiinni pylväässä. Lopuksi rakennettiin ja yhdistettiin kaksi muuta pylvästä, joista muodostui torni. Valmis torni on muodoltaan laatikkomainen.

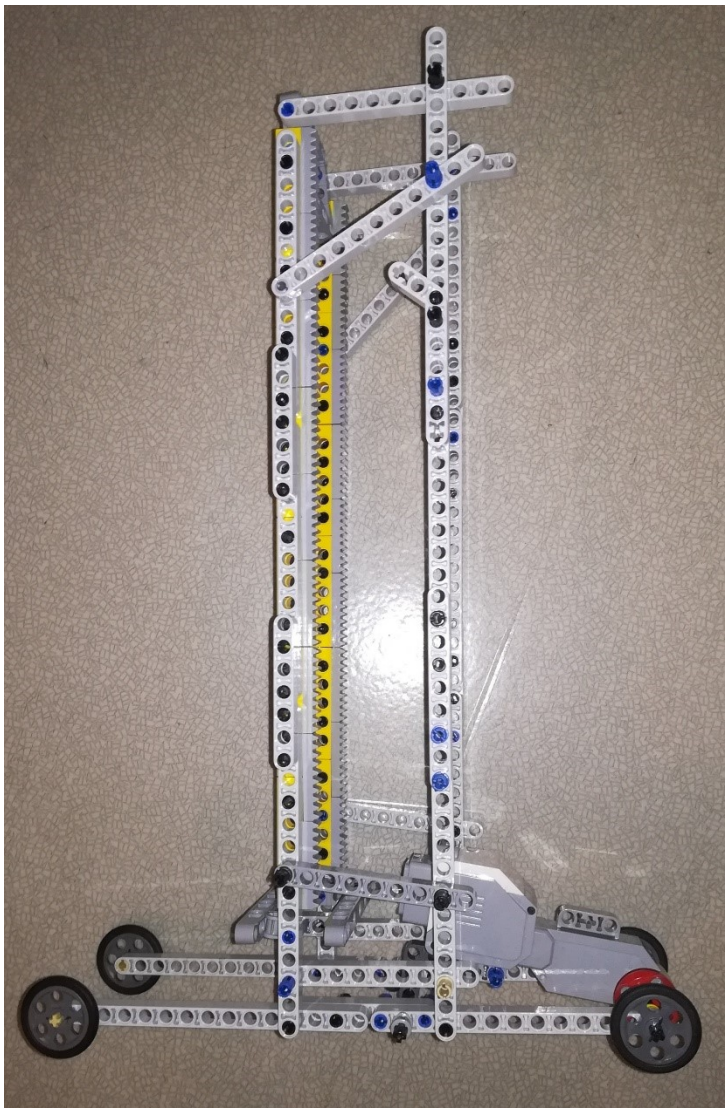
Testatessa tornin liikkuvuutta huomattiin moottorin kiinnityksessä haasteita. Moottori irtosi jatkuvasti, koska vastakappale oli huonosti asennettu. Torni piti purkaa monta kertaa ja koota uudestaan. Lopulta löytyi asento vastakappaleelle, jossa moottori liikkuu vapaasti. Tämän jälkeen moottori ei irronnut. Liukuhihna kiinnitettiin pystyliikkeen moottoriin. Ison moottorin vastakappale ei ole liian tiukka tai löysä.

Kun torniosuuden saatiin valmiiksi, yhdistettiin torni ja liukuhihnaosuus toisiinsa. Liukuhihna ja keskisuuri moottori asennettiin siten, että robotin massakeskipiste on mahdollisimman keskellä torniosuutta. Tämä sen vuoksi, että robotti ei kaadu helposti. Tällä on myös vaikutusta työturvallisuuteen. Asentamisessa oli haasteita. Keskisuurelle moottorille ei heti löytynyt parasta mahdollista paikkaa. Moottori saatiin asennettua niin, että robotti ei kaadu itsestään. Ulkoisen voiman vaikutuksesta se saattaa kaatua.

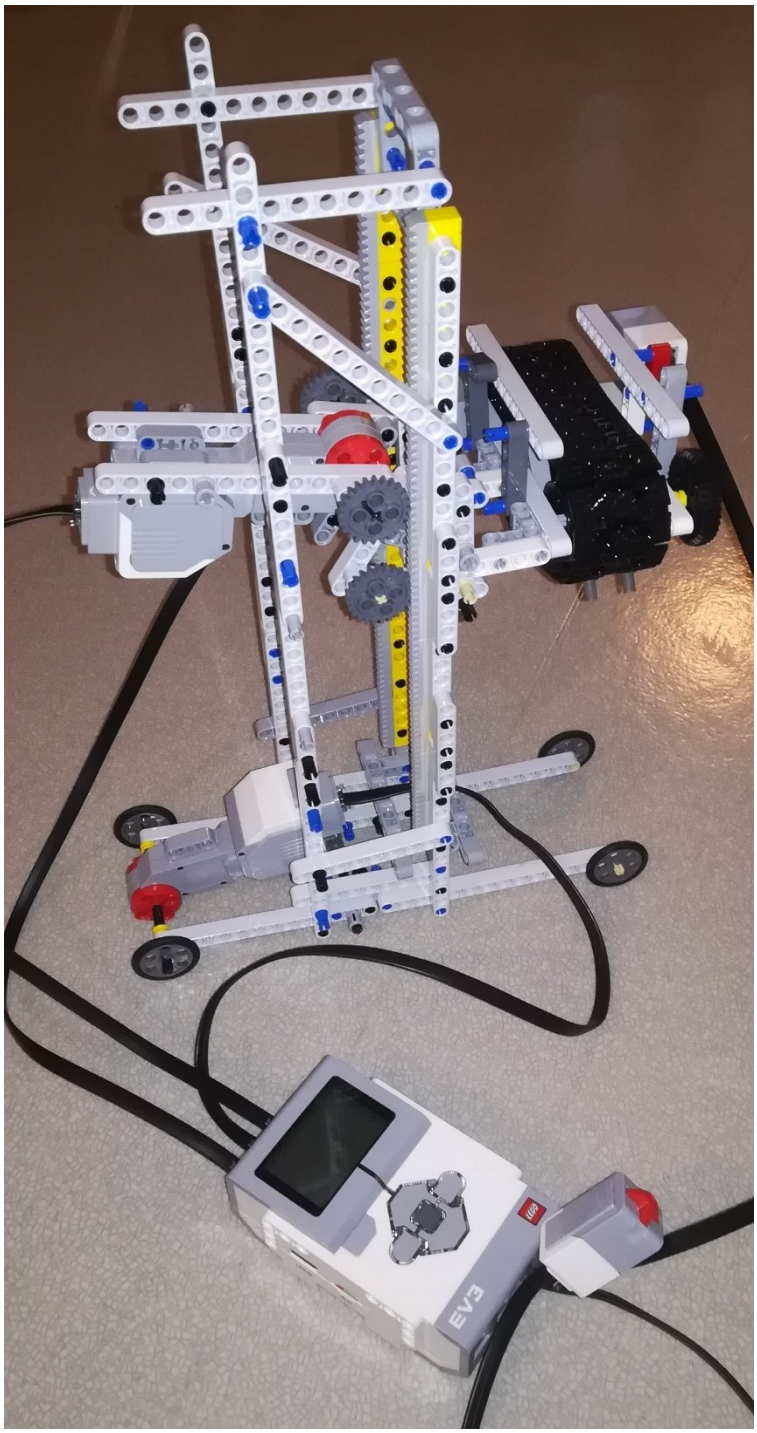
Seuraavaksi tehtiin robotin viimeinen osuuden, palkit, joihin kiinnitetään iso moottori ja sivuttaisliikkeen tekevät pyörät. Käytettiin isoa moottoria, joka kiinnitettiin torniosuuden pohjaan. Moottorin tarkoitus on liikuttaa pyöriä, jotka ovat tekevät robotin sivuttaisliikkeen. Pyörät kiinnitettiin palkkien päätyihin. Niitä on yhteensä neljä kappaletta. Iso osa robotin massasta koostuu pystysuoran liikkeen moottorista ja liukuhihnaosuudesta. Tämän vuoksi palkeista piti rakentaa pitkiä, jotta robotti ei kaatuisi. Testatessa robottia huomattiin, että robotti kaatuu helposti sivusuunnassa, jos se pysähtyy yhtäkkiä. Palkkien pituutta lisättiin kaatumisen estämiseksi. Robotti pysyy pystyssä, kun moottorien nopeudet asetetaan sopivan pieniksi.



Kuva 10. *Nostoliikkeen osuus, liukuhihna ja värisensori kiinnitetty yhteen*



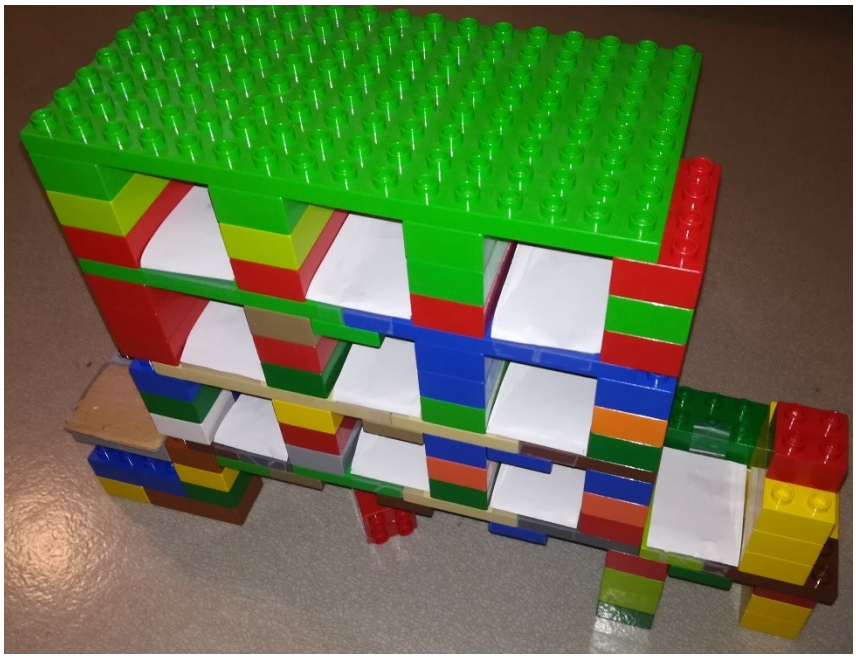
Kuva 11. *Robotin torni- ja sivuttaisliikkeen osuudet kiinnitetty toisiinsa*



Kuva 12. *Valmis robotti*

Hyllykön rakennusmateriaaleina käytettiin Lego Duplo-rakennuspalikoita. Lokerot rakennettiin tilaviksi, jotta robotti voi helposti asettaa laatikon lokeroon. Aiemmin mainitut kalibrointi- ja vientipaikat rakennettiin niille suunnitelluille paikoille. Kalibrointipaikkaan kiinnitettiin teipillä pahvinpalanen, jotta liukuhihnan koukut ottaisivat niihin kiinni. Robottia testatessa huomattiin, että robotti ei pysy asetetussa paikassa korkeussuunnassa silloin, kun robotti pysähtyy ylimmän rivin kohdalle. Tämän takia hyllykön ylin rivi poistettiin. Näin hyllykköön jäi kolme riviä. Hyllykön korkeutta korotettiin yhdellä palikalla,

koska liukuhihna ei päässyt tarpeeksi alas ottamaan laatikkoa alimmalta riviltä. Duplopalikoissa on kohoumia, jotka estävät laatikoiden liikuttamisen lokeroihin. Tästä syystä jokaisen lokeron pohjaan kiinnitettiin paperinpalaset teipillä.



Kuva 13. *Varaston hyllykkö*

Duplon keskipitkiä palikoita käytetään laatikkoina, joita robotti siirtelee hyllykössä. Palikoihin tehtiin rautalangasta kahvat. Teipillä saatiin rautalanka pysymään paikoillaan. Laatikoita valmistettiin 2 kappaletta. Liukuhihnan sivulle rakennettiin suojat, jotta laatikko ei putoa kyydistä. Robotti testatessa huomattiin, että pystysuunnan moottori ei jaksanut kannatella itseään tai liukuhihnaa tornin yläosassa. Tästä syystä hyllyrivejä piti pienentää yhdellä.

Kuva 14. *Hyllykössä käytettävä laatikko*

4.3.2 Ohjelmiston toteutus

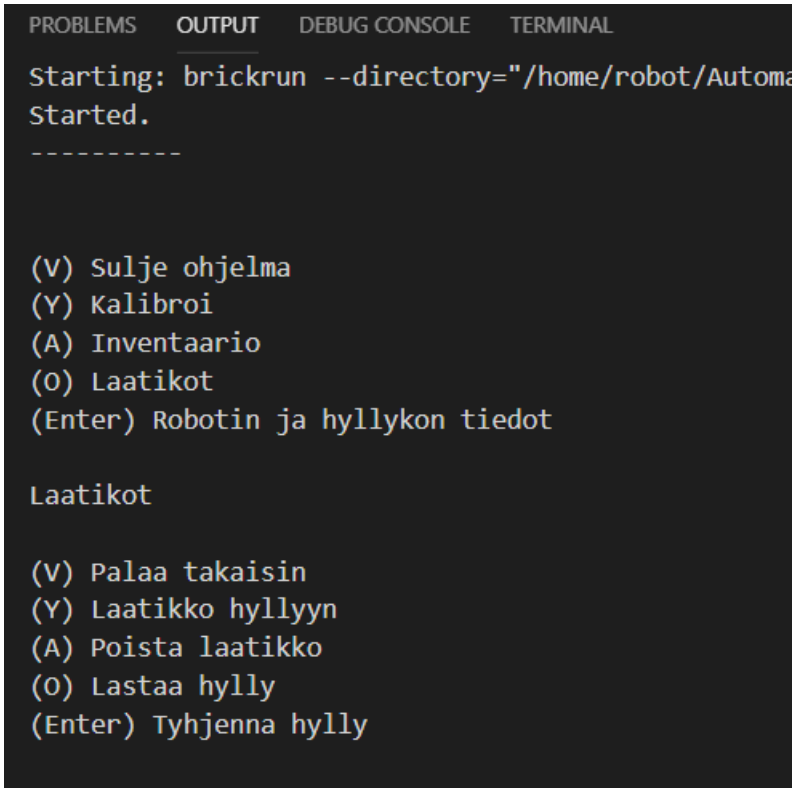
Moottorien, painikkeiden ja antureiden toiminnallisuutta testattiin. Jokaista moottoria ja anturia kokeiltiin erilisillä ohjelmilla, jotka tehtiin ainoastaan testejä varten. Käytiin läpi moottorien nopeuksia, tarkkuuksia ja sitä miten moottorit käyttäytyvät, kun niille aiheutetaan vastusta. Ohjelmiston toteutukseen käytettiin kirjaston vanhaa versiota. Tämä aiheutti sen, että dokumentaatio ei ollut yhtäpitävä kirjaston komentojen kanssa. Kirjasto päivitettiin uusimpaan versioon. Päivitettyä kirjastoa testattiin moottoreilla ja painikkeilla. Dokumentaatiossa mainitut komennot toimivat nyt niiden mukaan.

Suunnitelmien pohjalta tehtiin kaikkien luokkien, paitsi Hätästop-luokan, rajapintoja niin pitkälle, että luokkien funktioiden kutsuminen on mahdollista. Alussa ei lisätty toimintoja funktioille. Luokkaa Varasto_main toteutettiin niin pitkälle, että se pystyy kutsumaan muiden luokkien funktioita. Seuraavaksi alettiin tehdä Hyllykkö-luokkaa.

Seuraavaksi alettiin toteuttaa miten käyttäjä voi antaa komentoja robotille. Kuten aiemmin on mainittu, ohjelman käynnin aikana tietokoneelta ei voi antaa komentoja keskusyksikölle. Joten päädyttiin käyttämään keskusyksikön omia painikkeita. Painikkeet ohjelmointiin siten, että ohjelma tarkastaa 0,10 sekunnin välein painikkeiden tilan. Testatessa kävi ilmi, että tätä pienemmällä ajanjaksolla ohjelman valintoja tuli monta peräkkäin yhdellä painalluksella. Pienen aikavälin takia ohjelma luulee, että samaa painiketta painetaan monta kertaa, vaikka sitä painettaisiin vain kerran. Tällöin robotille tulee vääriä käskyjä. Tämän takia valittiin isompi aikaväli. Painikkeet lopulta ohjelmointiin niin, että ohjelma tarkastaa painikkeiden muutokset. Nyt ohjelma tarkastaa painikkeet 0,01 sekunnin välein. Jokaisen valinnan jälkeen on 0,10 sekunnin tauko. Painikkeita varten tehtiin olio, jonka avulla Painike-luokka muistaa mitä painiketta on painettu. Painikkeiden ja Varasto_main-luokkien toiminnallisuutta testattiin. Näiden testien jälkeen alettiin toteuttaa Robotti-luokkaa.

Hyllyköstä mitattiin tarvittavat mitat, joita ovat hyllykön korkeus ja leveys sekä lokeroiden mitat ja etäisyydet toisistaan. Mittauksissa käytettiin 20 cm:n viivoitinta ja kolmen metrin mittanauhaa. Testattiin kuinka monta astetta suurien moottorien tulisi liikkua jokaista liikuteltavaa senttimetriä kohden. Sivuttaissuunnan moottori pyöri kolme kierrosta. Robotin kulkema matka mitattiin ja laskettiin suhdeluku. Sama toistettiin pystysuunnan moottorille. Testattiin, kuinka monta astetta keskisuuren moottorin on liikuttava, jotta liukuhihna tekee täyden kierroksen. Käytettiin kahta eri koordinaatistoa. Aiemmin mainittu rs-koordinaatistoa käytetään lokeroiden paikantamiseen. Hyllykön mittauksien avulla luotiin xy-koordinaatisto robottia varten. Moottorien mittauksien avulla pystytään laskemaan xy-koordinaatiston pisteet ja pisteiden väliset etäisyydet asteina. Näillä tiedoilla robotti osaa liikkua lokeroiden välillä, kun parametrinä on halutun lokeron koordinaatit.

Kuvassa 15 näkyy ohjelman käyttöliittymä. Siinä näkyy, että käyttäjä on valinnut laatikoiden käsittelyn painamalla keskusyksikön oikeaa painiketta. Näyttöön tulostui käyttäjän tekemä valinta ja eri vaihtoehdot laatikoiden käsittelyä varten. Suluissa olevat tekstit kertovat, mitä painiketta painamalla saadaan valittua eri komentoja.



Kuva 15. *Ohjelman käyttöliittymä*

Värisensori kertoo, jos se ei saa mitattua väriä. Sensoria testatessa huomattiin, että välillä laite mittaa tyhjän lokeron kuin siinä olisi musta laatikko. Tästä syystä jätetään musta väri pois laatikoiden väreistä. Tämä väärinmittauksen mahdollisuus huomioidaan myös Hyllykkö-luokassa.

4.4 Työn haasteet

Tässä luvussa käsitellään haasteita, joita ilmeni robotin rakentamisessa sekä ohjelmiston toteuttamisessa. Osa ongelmista saatiin ratkaistua. Haasteista opittiin robotin rakentamiseen liittyviä asioita sekä miten käyttää kirjastoa niiltä osin, mikä liittyi työn toteuttamiseen. Osa ongelmista johtui fyysisten resurssien puutteista ja käytetyn kirjaston virheistä.

4.4.1 Suunnittelun haasteet

Ensimmäisenä haasteena oli se, että tekijällä ei ole aikaisempaa kokemusta Lego Mindstormsista. Internetistä etsittiin erilaisia toteutuksia robotille, muun muassa videoita katselemalla. Videoista saatiin tietoa Lego Mindstormsin mahdollisuuksista. Toiseksi tekijällä

ei ole aikaisempaa kokemusta käytetystä kolmannen osapuolen kirjastosta. Kirjastoon tutustuttiin internetin avulla selailemalla aiheeseen liittyviä videoita ja lukemalla esimerkkikoodeja. Koerobotin avulla selvisi, kuinka kirjaston avulla pystyy luomaan suurempia ohjelmia.

Hätäpysäytykseen suunniteltu tarkistussilmukka olisi pitänyt lisätä robotin jokaiseen moottorin liikkeeseen. Näin hätäpysäytys olisi saatu toimimaan joka tilanteessa. Tästä johtuen Robotti-luokan koodista olisi tullut hyvin pitkä ja toistuva. Suunnittelun aikana ei saatu selvitettyä korjausta tarkistussilmukan toistolle. Edellä mainittujen ongelmien vuoksi hätäpysäytys-toiminto jätettiin toteuttamatta.

4.4.2 Robotin haasteet

Keskisuuren moottorin sijoituksessa oli vaikeuksia, kun sille ei löytynyt parasta mahdollista paikkaa. Keskisuuren moottorin sijoituksen takia robotin painopiste on etäällä robotin keskustasta. Tämä tekee robotin asennosta epätasapainoisen, se on hutera ja kaatuu helposti ulkoisen voiman vaikutuksesta. Moottorit laitettiin liikkumaan suunniteltua hitaammin, jotta robotti ei kaadu liikkeen aikana. Rakennussarjassa oli rajallinen määrä legoja, joten suunnittelussa täytyi tarkkaan laskea tiettyjen palasten lukumäärä.

Pystyliikkeen moottori ei jaksanut kannatella kuormaa robotin yläasennossa, vaan hivuttautui alaspäin. Tämä korjattiin poistamalla ylin rivi hyllyköstä. Myös robotin sivuttaisliikkeen kanssa oli ongelmia. Jos robottia ei asetettu täysin kohdakkain hyllykön kanssa, robotti liikkui poispäin hyllyköstä tai törmäsi helposti siihen. Tätä ei saatu korjattua. Robotin antureita ei ole tarkoitettu tarkkaan käyttöön, mikä teki mittaustuloksista ja moottorin liikkeistä epätarkkoja.

4.4.3 Ohjelmiston haasteet

Ohjelman ja robotin testaus oli hidasta, koska debuggaus ei onnistunut valitulla kirjastolla. Ohjelman ja robotin eri vaiheiden tietoja tulostettiin testausta varten. Ainoastaan Hyllykkö-luokkaa pystyi testaamaan debuggerilla, koska luokassa ei käsitelty Lego Mindstormsin antureita tai toimilaitteita.

Yksi haasteista oli, kun moottoreita käskettiin liikkumaan alle kymmenen astetta, ne jäivät paikoilleen. Tämän aiheutti sen, että ohjelma ei voinut jatkaa toimintaansa, koska moottorit eivät päässeet tavoitearvoonsa. Tämä ongelma ratkaistiin asettamalla moottoreiden toiminnoille aikakatkaisu. Aikakatkaisun pituudeksi laitettiin 5,0 sekuntia. Jos moottori ei ole päässyt tavoitearvoonsa tähän aikaan mennessä, moottori pysäytetään ja ohjelma jatkaa etenemistään. Tämä aiheuttaa sen, että robottia ei aina saada tarkalleen haluttuun paikkaan.

4.5 Lopputulos

Lopputuloksessa pohditaan työn onnistumista, asetettujen vaatimusten täyttymistä sekä mahdollisia jatkokehitysideoita. Kaikkia vaatimuksia ei saatu täytettyä. Jatkokehitysideoita pohtiessa huomattiin, että järjestelmän parantamiseksi tarvittaisiin lisätä sensoreita ja rakennuspalikoita sekä ohjelmoida uusia luokkia.

4.5.1 Sovelluksen onnistuminen

Käyttöliittymä on selkeä ja komentojen syöttäminen ei tuota ongelmia. Ohjelma pitää kirjaa robotin ja hyllykön tiedoista. Robotti on jaettu selkeisiin osiin, ja ne on saatu yhdistettyä yhdeksi kokonaisuudeksi. Robotti saatiin suuremmaksi osaksi rakennettua rakennussarjan palikoilla. Hammasratas palikoita piti hankkia erikseen. Järjestelmä suoriutuu peräkkäisistä toiminnoista, eikä jää jumiin rakenteellisista tai ohjelmallisista syistä. Vikatilanteita tuli esiin eri toiminnoissa, mutta niitä saatiin vähennettyä.

Järjestelmän ohjelma on suunniteltu huolellisesti ja sen toteutus vastaa suunnitelmaa. Luokkajako on selkeä, joten uusien ominaisuuksien lisääminen on helppoa. Koodi on kommentoitu, joten muun kuin tekijän on helppo tarkastella ohjelmaa. Kaikki luokat, paitsi Robotti ja Hätästop, saatiin valmiiksi. Robotti-luokka ei käsittele laatikoita, eli robotti ei pysty ottamaan laatikoita kyytiin tai poistamaan niitä. Hätästop-luokkaa ei lähdetty ollenkaan toteuttamaan. Nämä asiat tarkoittavat, että järjestelmä ei suoriudu kaikista käyttötapauksista, ja robottia ei voida pysäyttää hätäpysäytyspainikkeella. Ohjelma osaa laskea robotin viimeisestä liikkeestä aiheutuneen laskennallisen virheen. Tätä laskua käytetään virheasentojen vähentämiseen.

4.5.2 Vaatimusten täyttyminen

Robotti ei toteuta kaikkia sille asetettuja vaatimuksia. Laatikoiden käsittely jäi ohjelmointipuolella kesken, joten robotti ei pysty ottamaan laatikkoa kyytiin tai poistamaan sitä kyydistä. Tämän takia robotti ei voi siirtää laatikoita. Myös turvallisuusvaatimuksien täyttämisessä oli puutteita. Hätäpysäytystoimintaa ja hälytyksiä ei toteutettu.

Moni muu vaatimus saatiin täytettyä. Robotti osaa kalibroida itsensä, suorittaa inventaarion varastosta sekä liikkua lokeroiden välillä suhteellisen pienillä virheillä. Robotti tunnistaa laatikon tämän värin perusteella. Ohjelma näyttää käyttäjälle robotin ja hyllykön tietoja. Ohjelma ottaa vastaan käyttäjän antamat komennot, hätäpysäytys poissulkien. Varasto ja käyttäjä on erotettu toisistaan pitkien johtojen avulla.

Ohjelmoinnissa käytettiin puhtaasti Pythonia ja ohjelman suunnittelussa UML:ää. Ohjelmaa voi käyttää vain yksi henkilö kerrallaan. Robotti on helppo purkaa osakokonaisuuk-

siin ja sen jälkeen helppo koota takaisin. Robotti kertoo käyttäjälle toiminnan onnistumisesta. Ohjelma päivittää automaattisesti hyllykön tietoja. Käyttöliittymä on selkeä ja ohjelman käyttö ei vaadi erillisiä ohjeita.

4.5.3 Jatkokehitys

Työtä tehdessä huomattiin mahdollisia parannus- ja jatkokehitysideoita. Robottiin voidaan rakentaa kiskot sivuttaisliikettä varten. Tällä saadaan estettyä robotin kääntyminen hyllyköstä poispäin. Rakentamiseen käytetään samoja palikoita, joita käytettiin robotin tornin rakentamisessa. Näin saadaan rakennettua paikallaan pysyvät kiskot. Lisäksi renkaat korvataan hammasrattailla. Robottiin voidaan rakentaa kappaleita, jotka estävät robotin heilumisen. Kappaleita ovat palkki ja kisko. Palkki kiinnitetään hyllykön päälle tehtyyn kiskoon.

Työssä oli suunniteltu, että hätäpysäytyspainikkeen toimintaa seurataan robotin liikkeiden ja painikevalintojen aikana. Kirjastoa käsittelevissä ohjeissa [11] neuvottiin säikeiden käytöstä. Yhden säikeen avulla pystytään seuraamaan hätäpysäytyspainikkeen toimintaa sekä suorittamaan muuta ohjelmaa samanaikaisesti. Hätäpysäytyspainiketta painettaessa täytyy ottaa huomioon, että robotin mahdollisesti jo keräämät hyllykkötiedot saadaan talteen. Törmäyksenestoa voidaan kehittää käyttämällä kosketusantureita.

Käyttöliittymä koostui keskusyksikön painikkeista ja tietokoneen ruudusta. Kirjasto tarjoaa rajapinnat keskusyksikön LCD-näytön hallitsemiseen. Näytölle voidaan tulostaa tekstiä ja mustavalkoisia kuvia. Näyttö voi korvata tietokoneen näytön. Näin saadaan käyttöliittymä kokonaan samaan laitteeseen. Testatessa huomattiin, että liukuhihnalla oli vaikeuksia saada laatikko kyytiin. Liukuhihnaan voidaan kiinnittää joko väri- tai kosketussensori tarkistamaan, saatiinko laatikko kyytiin. Tarkistus- ja korjaustoiminnat pitää ottaa ohjelmassa huomioon. Luokka Varasto_main sisältää tietojen tulostuksen käyttäjälle. Tietojen esittämisen voi erottaa omaksi luokaksi.

5. YHTEENVETO

Tässä työssä tarkasteltiin Lego Mindstorms EV3:n soveltavuutta pienten robottisovelluksien tekemiseen. Kiinnostuksena oli tehdä automaatioon liittyvä robotti. Lego tarjoaa graafisen ohjelmointiympäristön robotin ohjelmointiin, mutta valitettavasti tämä on rajallinen olio-ohjelmoinnin kannalta. Tässä työssä käytettiin kolmannen osapuolen kirjastoa ev3dev. Kirjasto tarjoaa kattavat rajapinnat antureille, keskusyksikölle ja toimilaitteille. Se ei kuitenkaan ole virheetön, sillä pieniä heikkouksia esiintyi testauksen ja koodaamisen aikana. Pythonia on helppo käyttää. Kirjastosta löytyi versio, jolla sitä voidaan käyttää valitulla ohjelmointikielellä.

Työssä määritettiin automaation asettamia vaatimuksia. Ohjelman suunnittelussa otettiin huomioon määritetyt vaatimukset. Robottia voidaan ohjata toteutetun käyttöliittymän kautta. Käyttöliittymään tulostuu komentoja, joita käyttäjä voi antaa. Siihen tulostuu myös tapahtumien ja hyllykön tietoja. Suurin osa asetetuista vaatimuksista toteutui. Suunnitelmien pohjalta saatiin toimiva järjestelmä, jossa on kuitenkin puutteita. Järjestelmä ei suoriudu kaikista käyttötapauksista. Antureiden ja toimilaitteiden epätarkkuuksien vuoksi kaikkia virhetilanteita ei saatu poistettua.

Legon rakennuspalikoiden ansiosta robotin rakenne ja muoto oli helppo suunnitella sekä toteuttaa. Työssä käytetty kirjasto osoittautui hyödylliseksi välineeksi ohjelmoinnissa. Selkeä dokumentaatio ja hyvät koodiesimerkit auttoivat ohjelman toteutuksessa. Kirjastossa on myös helposti käytettävät rajapinnat keskusyksikölle, antureille ja moottoreille. Kirjastosta löytyy kuitenkin virhe, jonka takia moottori ei liiku ollenkaan, jos sen halutaan liikkuvan vain muutaman asteen. Toteutetussa ohjelmassa on selkeä luokkajako, joten uusia ominaisuuksia voidaan lisätä ja vanhoja muuttaa. Työn lopussa käytiin läpi järjestelmän parannusehdotuksia.

LÄHTEET

[1] K.J Roodbergen, I Vis, A survey of literature on automated storage and retrieval systems, European Journal of Operational Research, Vol. 194, No. 2, 2009, s.343–362.

[2] G. Jinxiang, M Goetschalckx, L.F McGinnis, Research on warehouse operation: A comprehensive review, European Journal of Operational Research, Vol. 177, No. 1, 2007, s.1–21.

[3] S. Haddadin, A. Albu-Schäffer, G. Hirzinger, Safety Analysis for a Human-Friendly Manipulator, International Journal of Social Robotics, Vol. 2, No 3, 2010, s.235–252.

[4] A.H. Mujtaba, Lego Mindstorms EV3 Essentials, Packt Publishing, 2014.

[5] İ. Kunduracıoğlu, Examining the interface of lego mindstorms Ev3 robot programming, Journal of Educational Technology and Online Learning, 2018, s.28 – 46.

[6] W. Burnett, Nine alternative programming languages for Lego Mindstorms, Lego Engineering, verkkosivu Saatavissa (viitattu 2.2.2019): http://www.legoengineering.com/alterna-tive-programming-languages/

[7] ev3dev, verkkosivu Saatavissa (viitattu 2.2.2019): https://www.ev3dev.org/

[8] Python language bindings for ev3dev, verkkosivu Saatavissa (viitattu 10.2.2019): https://python-ev3dev.readthedocs.io/en/ev3dev-stretch/index.html, Copyright 2015, Ralph Hempel et al.

[9] N. Ward, EV3dev & EV3 Python, verkkosivu Saatavilla (viitattu 18.5.2019): https://www.yo-tube.com/watch?v=BDr3lDmhkOQ&list=PL7hndBcWv-umf5tdIp0lJfPuU0X9LE97a

[10] Brickself, Model of automatized warehouse made using LEGO Mindstorms EV3 platform, verkkosivu Saatavissa (viitattu 4.6.2019): http://www.brick-shelf.com/cgi-bin/gallery.cgi?f=568743

[11] N. Ward, EV3 Python, verkkosivu Saatavissa (viitattu 20.6.2019): https://sites.google.com/site/ev3devpython/